Route Guidance System Based on Self-Adaptive Algorithm

Mortaza Zolfpour-Arokhlo, Ali Selamat,
Siti Zaiton Mohd Hashim, and Md Hafiz Selamat

Faculty of Computer Science & Information Systems,
Universiti Teknologi Malaysia, 81310 UTM Skudai, Johor, Malaysia
zolfpour@gmail.com, {aselamat,sitizaiton,mhafiz}@utm.my

Abstract. Self-adaptive systems are applied in a variety of ways, including transportation, telecommunications, etc. The main challenge in route guidance system is to direct vehicles to their destination in a dynamic traffic situation, with the aim of reducing the motoring time and to ensure an efficient use of available road resources. In this paper, we propose a self-adaptive algorithm for managing the shortest paths in route guidance system. This is poised to minimize costs between the origin and destination nodes. The proposed algorithm was compared with the Dijkstra algorithm in order to find the best and shortest paths using a sample simplified real sample of Kuala-Lumpur (KL) road network map. Four cases were tested to verify the efficiency of our approach through simulation using the proposed algorithm. The results show that the proposed algorithm could reduce the cost of vehicle routing and associated problems.

Keywords: Traffic control, route guidance system (RGS), self-adaptive, shortest path problem (SPP), Dijkstra's algorithm, Urban road network(URN).

1 Introduction

Due to the variety of road network and environment the corresponding route guidance systems are becoming more complex and present many new characteristics such as traffic congestion, time to travel, etc. Therefore, the vehicle driver needs to use a route guidance system in order to reduce the traffic delays. Nowadays, there are various successful algorithms for finding the shortest path in the route guidance system. One of the most popular of these types of algorithms is Dijkstra algorithm. The vehicle drivers with different available solutions need a quick response to the change in road network conditions. The main challenge faced by the route guidance system is directing the vehicles to their destination in the dynamic traffic situation, with the aim of reducing travel times and efficient use of available network capacity. Therefore, the vehicle driver needs to use a route guidance system based on self-adaptive algorithms. Any Change in the environment can be perceived by the self-adaptive algorithm because the self-adaptive system is a complex system as it has the ability to change structure and behavior itself [1], [2]. Generally, to solve the problems, using a fast path planning method seems to be an effective approach to improve the route guidance system. The main thrust of this is to compare the

D. Lukose, A.R. Ahmad, and A. Suliman (Eds.): KTW 2011, CCIS 295, pp. 244–253, 2012.

shortest path algorithms with Dijkstra algorithm. The shortest path (i.e. lowest cost) is calculated using the proposed algorithm, and it will be suggested to vehicle drivers in each intersection (or node). Dijkstra algorithm is a search graph for routing problems and producing a graph of the shortest path in a graph [3]. This algorithm finds the shortest path (i.e. lowest cost) between origin and destination nodes in a directed graph. For example, Dijkstra algorithm can find the shortest path (or lowest cost) between one points of the city to another point of city in a trip [4]. The shortest path problem has been investigated extensively, and many modern algorithms have been proposed to solve this problem. Yet, many of the existing algorithms fail to meet some of the requirements of the solutions applicable in the real world. Therefore, this paper proposes a method to solve the path planning problem in route guidance systems in terms of accuracy and speed. The results obtained from the approach have been compared with Dijkstra algorithm [3]. The rest of the paper is organized as follows. In Section 2, we start with an overview of path planning problem, and some basic concepts related to the study. In Section 3, we propose a self-adaptive algorithm for route guidance system, a formal problem formulation and dynamic shortest path algorithm in route guidance system. In section 4, experimental comparison and simulation, results are analyzed with some samples. The conclusion and some future works are discussed in the last section.

2 Related Works

In this section, we present the essential background for our contribution. A brief review of related studies is given with particular dynamic shortest path algorithms focused on a route guidance system. A new approach in the urban road network, which can reserve time and space for cars at an intersection or junction has been studied in [6]. A shortest path problem is used on finding the path with minimum travel distance, time or cost from one or more origins to the one or more goals or destinations through a connected network [4]. It is an important issue because of its wide range of applications in transportation, robotics and computer networks. The shortest path from one node to every other node can be found by using a number of various algorithms [5]. Dijkstra algorithm is probably the best-known and most important, popular, and successfully implemented shortest path algorithms for both lecturers and students in computer science [7]. Therefore, road traffic control flow through the use of the self-adaptive algorithm for route guidance system in dynamic transportation is one of the reasons for this paper. However, most route guidance system based applications are concentrated on modeling and simulation. Hence, new efforts and researchers should be done for real-world applications. In this research, we study a route guidance system framework for solving urban road traffic problems.

2.1 Review of the Self-Adaptive System

Over the past decade, numerous frameworks have been done for the successful implementation of self-adaptive systems [8]. Several systems properties can be controlled using self-adaptive systems [9] like self-healing system, which can automatically diagnose, correct faults, and discover. In the alternative, a

self-optimizing system can automatically monitor and adapt resource usage to ensure optimal functioning relative to defined requirements [10].

2.2 Review of the Dijkstra algorithm

Dijkstra algorithm is widely used and is a successful algorithm for routing problems to find the shortest path in a weighted directed graph network graph [3]. It is often used in routing. As a result, the shortest path first is widely used in network routing protocols.

3 Proposed Route Guidance System

The proposed algorithm can also be used for shortest path and finding costs in the urban road network. A route guidance system (RGS) will calculate the shortest path from current vehicle position to destination locations, so that the traveling cost or time for drivers will be minimized. We have used shortest path computation (SPC) to calculate the nearest distance of the destinations with the support of vehicles in the route guidance system. The updated information of the vehicle will be used by the system to find the shortest path from current vehicle position to destination zone for drivers based on current conditions. However, one of the requirements of intelligent transportation system dynamic is real and current information about travel time for a continuous path that can be acquired by several detectors such as magnet sensors, video cameras, GPS, GSM (global system for mobile communication) and also other network traffic sensors on the routes to transport [12]. We have applied the roles of path planning algorithm in route guidance system as follows:

- Transferring information acquired through the sensors;
- Receiving a route request from vehicles;
- The shortest path computation and sending it to vehicle's drivers;

3.1 Problem Formulation and Properties

A road directed graph presented by G=(V,A) is a directed dynamic route guidance system based on an electronic map, a set of '*N*' nodes (V) and '*M*' directed edges (A). Each $R_{(s,d)}$ edge is a nonnegative number which stands for the cost while 's' is a start node and '*d*' is a finish node connected to the $R_{(s,d)}$. Consider a directed graph, G, which is composed of a set of '*s*' nodes and a set of '*r*' directed edges. Set a number of edges as cost(C) table. Furthermore, G file is filled in data file. If S= { s_1, s_2, ….., s_n }, D= { d_1, d_2, ….., d_n } then $R_{(s,d)}$ = { $R_{(s_1,d_1)}$, $R_{(s_2,d_2)}$, …., $R_{(s_n,d_n)}$ }

$R_{(s,d)}$ consists of the sum of all edge distances (costs) participating in the network path. Therefore, according to the trip origin (s) node and destination (d) node, this issue can be solved as the optimization problem based on real transportation network that is defined as follows:

Fig. 1. A part of Kuala Lumpur city road network map

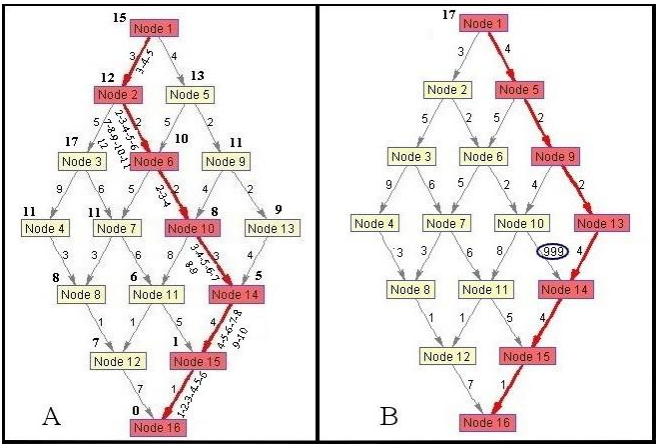

Fig. 2. G (16,24) is the optimal path in ideal (A) and blocked(B) routes status in Case 1, respectively

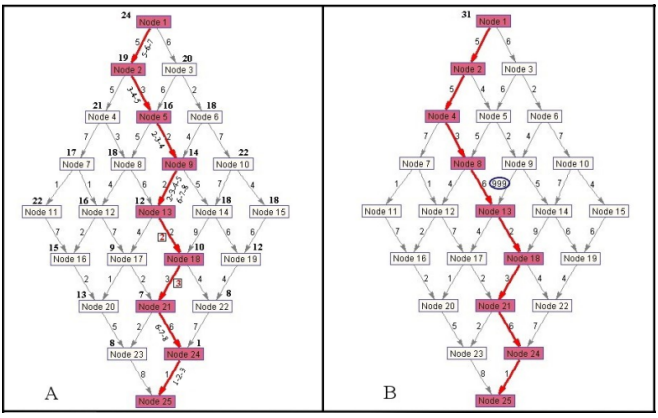

Fig. 3. G (25,40) is the optimal path in ideal (A) and blocked(B) routes status in Case 2, respectively

$$R^{*}_{(s,d)} = \min \sum_{i=0}^{d} \sum_{j=0}^{d} a \; R_{(s_j,d_j)} \begin{cases} a = 1 & A\ link\ exists\ between\ i\ and\ j \\ a = 0 & Otherwise \end{cases} \quad (2)$$

Where:

— i and j are current state movement into right and bottom side directions.
— $R_{(s,d)}$ is the shortest path from a origin node, 's' to a last node, 'd'.
— α is binary digits (0 or 1).
— Links are independent of each other.

However, as mentioned in the previous section real-time information systems can be acquired using video cameras, GPS and other devices on transportation routes. In this paper, travel cost is assumed to be random variable [13].

3.2 Route Guidance System Based on Self-Adaptive Algorithm (RGSBSAA)

Figure 1 shows the road network map of a part of Kuala Lumpur (KL), the capital of Malaysia. A grid network of Figure 1 has been drawn in Figure 2A. Proposed algorithm (RGSBSAA) can also be used for finding shortest path (i.e. lowest cost) from origin node (i.e. node 1) to destination node (i.e. node 16) in a road network graph. The input of the algorithm consists of distances directed between nodes as r and a source data from origin and destination paths as s, d in the network graph G. If the distance between two nodes can be defined as path cost, the total cost of a network graph is the sum of all route distances between nodes in a graph. This algorithm is expressed based on node weight. Indeed, in this study, a "*node weight*" is defined as the shortest distance from each node to last network node. If "n" is the last node number, "*n-1*" and "k" are two node numbers connected to "node n" in the network. And also, regarding equation (2) and node weight definition, the procedure is presented as follows: All node's weights in the road graph are calculated in two steps as follows:

- **Step 1**
  - Distance between node *n-1* and node n are set to the weight of node *n-1* and also distance between node k and node n is set to the weight of node k (expect for first and last nodes, each node has two input and one output routes or at least one input and one output route).
- **Step 2**
  - Similarly, the procedure continues to calculate weights of all next nodes until the first node weights is calculated. Finally, the weight of the first node will be the minimum distance paths between first points to last nodes in the network.

Now, consider calculated node weights in RGSBSAA algorithm as the input data file, the shortest path computation procedure is presented as follows:

- Transportation route is started from first node (node 1), then amount of each distance route connecting the first node minus the weight of first node. If the result number is equal to weights of the each node, then the second node is connected to first node and it will be a priority route of network or next optimal route.
- Procedure will continue along the route connected to the last node then the shortest path network is determined.

The above algorithm uses the listed assumptions as follows:

- The road network is one-way, and it is from left to right and towards bottom direction.
- There is the one-way directional connection between all nodes.
- The information of distance routes between all nodes exists before the start of the travel.
- The travel is based on electronic map.

RGSBSAA Algorithm: Rote Guidance System based on Self-adaptive Algorithm.

```
1.  INPUT:
2.    G( V, E) is a data file which composed a set of V nodes and set of E directed edges.
3.  PARAMETERS:
4.    R(s,d) , a nonnegative number stands for the cost where "s" is start node and "d" is last node.
5.    i, j , k; loop index, G( 1, i) is array of vertexes source; G( 2, i) is array of vertexes destination.
6.    G( 3, i) is array of edge distance(or cost); W(i) is array of node costs(node weights) table
7.    for each node and P(i) is array of shortest path from the origin to final node.
8.  OUPUT:
9.    W(k) is a cost data table for all nodes. P(k), a shortest path data table for a graph.
10.   S(j), cost limit s of shortest path routes.
11. INITIALIZATION:
12.   // All nodes from last to first nodes are examined for the routes connected nodes.
13.   // For each edge do operation in two steps as follows:
14.   set W[1... n-1] = 999 , W(n) = 0 , P(i) = 0 ;
15. BEGIN // step 1: Node weight computation
16.     for all nodes // for each node and edge pick the costs in W(k).
17.           for j = first to last edges // j is set to the destination node of edges.
18.                 if (G( 2 , j) = i ) // k is set to the source node of edges.
19.                       W(k) = W(i) + G( 3 , j);
20.                 end if
21.           end for
22.     end for
23.   // step 2: Shortest Path computation
24.     for i = first to last edges
25.           while (the origin (k) is the same in graph, G )
26.                 if (G( 3 , i) = W(k) - W(j))
27.                       P(k) = G( 2 , i);
28.                 else
29.                       i = i + 1;
30.                       k = G( 1 , i);
31.                 end if
32.           end while
33.     end for
34.   // step 3: Cost limit computation for self-adaptive
35.     for j = first to last edges // j is set to the destination node of edges.
36.           while (edge belong to p(k) & defined shortest path is true )
37.                       C(j) = G( 3 , j) + 1;
38.                       S(j) = { G( 3 , j), C(j) }; cost limit s of shortest path routes.
39.           end while
40.     end for
41. END
```

Table 1. Performance measure for weighted directed graphs on sparse graphs

Criterion	Description
BPC	Best path cost among all the RGSBSAA and Dijkstra runs
AvePC	Average path cost obtained for all RGSBSAA and Dijkstra
AvePCGap	Difference between the average path cost found (AvePC) and the best found path cost (BPC) of the test problem. AvePCGap = (AvePC - BPC)/BPC*100%

Table 2. Simulation results of RGSBSAA and Dijkstra algorithms for weighted directed graphs on sparse graphs

(V , E)	Method	Elapsed time	BPC	AvePC	AvePCGap
$G(16, 24)$	RGSBSAA	0.001124	17	17.00	0.00%
	Dijkstra	0.001211	19	19.00	0.00%
$G(20, 31)$	RGSBSAA	0.001210	27	27.00	0.00%
	Dijkstra	0.001215	26	26.00	0.00%
$G(20, 31)$	RGSBSAA	0.001214	25	25.10	0.40%
	Dijkstra	0.001281	27	27.35	1.30%
$G(20, 37)^*$	RGSBSAA	0.001282	131	131.50	0.38%
	Dijkstra	0.001315	135	136.80	1.33%
$G(20, 49)$	RGSBSAA	0.001294	145	145.00	0.00%
	Dijkstra	0.001341	151	152.00	0.66%
$G(46, 69)$	RGSBSAA	0.001232	162	163.35	0.83%
	Dijkstra	0.001377	155	156.20	0.77%
$G(46, 71)$	RGSBSAA	0.001263	164	165.30	0.79%
	Dijkstra	0.001370	158	159.50	0.95%
$G(100, 197)^*$	RGSBSAA	0.001422	850	852.00	0.24%
	Dijkstra	0.001534	810	815.20	0.64%
$G(100, 203)^*$	RGSBSAA	0.001411	862	865.00	0.35%
	Dijkstra	0.001538	843	860.30	2.05%
$G(100, 216)^*$	RGSBSAA	0.001446	1145	1164.00	1.66%
	Dijkstra	0.001548	1162	1188.46	2.28%
$G(100, 217)$	RGSBSAA	0.001433	1125	1141.30	1.45%
	Dijkstra	0.001559	1210	1231.00	1.73%

* In these types of graphs, there are more than two connected edges for some of nodes (i.e. 3 or 4 edges).

4 Comparison of Experimental Results

In this section, simulation experiments have been carried out on different network topologies for road networks consisting of 6 to 100 nodes with different edges (between 10 to 220 edges). In addition, Table 2 is simulation results of RGSBSAA algorithm and Dijkstra algorithm for weighted directed graphs on sparse graphs (i.e. road network graphs). BPC and AvePC report the shortest and average path cost found in the mentioned graphs in RGSBSAA algorithm runs respectively. Also, AvePCGap reports the measures of the gap between shortest path and average path cost. To summarize, for all graphs proposed in the Table, the average path cost gaps performance in RGSBSAA algorithm were obtained less than the Dijkstra algorithm performance.

4.1 RGSBSAA Algorithm - Case 1

Figure 2A shows the weight of node 1 is 15, and the weight of node 2 is 12... Finally the cost (or weight) of node 16 is 0 that is calculated with first part of proposed algorithm procedure. The cost (or weight) of 15 is the shortest distance from node 1 to last node (destination node). Figure 2A presents the optimal routes (the red color path) in ideal route status as follows: node 1 → node 2 → node 8 → node 10 → node 14 → node 15 → node 16. Figure 2B shows the route between node 10 and node 14 which is blocked (let cost= ∞) because of traffic congestion; therefore, some connected routes are re- moved automatically in the new optimal route computation and the next route which is connected node 1 (i.e. node 2) is selected as first choice route in this figure. Furthermore, Figure 2A shows the self-adaptive algorithm performance measures of the shortest path routes. Self-adaptive algorithm will provide specifications as to what costs exist for a particular route within a given time frame, after which an alternative route must be taken. The RGSBSAA algorithm uses the received traffic congestion information of shortest path routes from the traffic-control devices in each intersection with acceptable route critical costs suggested by the proposed algorithm in Algorithm RGSBSAA. For example, in first route of Figure 2A, the costs limit of node 8 to node 10, 2,3,4 show that the cost's limit encountered that vehicle driver can use the costs of 2,3 and 4 for taking this shortest path. The out of 2,3,4 range which causes are the change in the recommended shortest path. Therefore, consider the above-mentioned constraints and proposed RGS- BSAA algorithm, the optimal path in blocked route status (Figure 2B) is the red color path between node 1 and node 16. Last, consider that RGSBSAA algorithm and Figure 2B routes of node 10 and node 14 is congested (or blocked) therefore, its cost tends to ∞. Figure 2B shows the minimum cost (or distance) between node 1 and node 16 is changed from first minimum cost (in Figure 2A), 15 to 17 in Figure 2B.

4.2 RGSBSAA Algorithm - Case 2

Consider the previous section discussion; Figure 3A presents a transportation network that has 40 routes and 25 nodes as intersection. The distance between node 1 and node

25 (or last node) is 24 i.e weight of node 1. Figure 3A presents the optimal routes in ideal route status as follows: node 1→node 2→ node 5→ node 9→ node 13→ node 18→node 21→node 24→ node 25. Also, Figure 3A shows the self-adaptive algorithm performance measures of the shortest path routes. For example, in first route of Figure 3A, the costs limit of node 1 to node 2, 5,6,7 show that the cost's limit encountered that vehicle driver can use the costs of 5,6 and 7 for taking this shortest path. The out of 5,6,7 range which causes are the change in the recommended shortest path. Therefore, considering the above-mentioned constraints the proposed RGSBSAA algorithm, the optimal path in blocked route status (Figure 3B) is the red color path between node 1 and node 25. Lastly, consider that RGSBSAA algorithm routes of node 9 → Node 13 is congested (or blocked) therefore, their costs tend to ∞. Figure 3B shows the minimum cost (or distance) between node 1 and node 25 is changed from first minimum cost in Figure 3A, 24 to 31 in Figure 3B. However, the simulation program is tested with network graph information. The acquired results affirmed the proposed algorithm results were the convergence to acquire the shortest path in each network case. In order to assess the performance of self-adaptive algorithm, Table 2 shows the comparison of some of our cases experimental study with Dijkstra algorithm. We can see that the result of our proposed algorithm achieves better results than Dijkstra algorithm. This comparison in a real transportation network shows the evaluation of proposed method that is route guidance system based on a proposed path planning algorithm. Finally, in all experimental cases, average path cost gaps (AvePCGap) of RGSBSAA method are less than average path cost gaps of Dijkstra algorithm and has a better performance result.

5 Conclusion and Future Work

This paper proposes a new self-adaptive system algorithm in route guidance system for finding the critical route costs and dynamically responds to the environmental conditions for any stated path. The behavior of the route guidance system change based on the proposed algorithm. The performance of the proposed algorithm was evaluated using adaptive routing agents that dynamically and adaptively find the minimum delay and maximize time between the routes in the route guidance system (see Table 2). Through simulation, the proposed algorithm (RGSBSAA) was applied for managing the shortest path routes for many network graphs with a number of edges ranging from 4 to 200(like Figures 2A and 3A). In each node (intersection), the shortest path (or lowest cost) was determined by the Dijkstra algorithm. Whenever the agent reports traffic congestion on in a specific route using the initial shortest path, the drivers take next shortest path based on the proposed algorithm. Given the above results, our contributions are made as follows:

- We have demonstrated a new self-adaptive algorithm to find the shortest path problem in the road network graph.
- It develops a novel approach to the route guidance system in the underlying network since the vehicle driver receives information from next road status he/she follows to the next shortest path.

– The study provides empirical grounds for the route guidance system based on self-adaptive algorithm that could perform well on network graphs.

Future works are suggested as follows: Vehicle route selection could be adapted online, i.e. vehicles could coordinate their behavior based on real-time traffic conditions in the road network. In the future study, many real-world factors in the environment should be considered such as vehicle accidents, weather, illegal parking, etc.

Acknowledgement. The authors wish to thank Ministry of Higher Education Malaysia (MOHE) under Fundamental Research Grant Scheme (FRGS) Vot 4F031 and Universiti Teknologi Malaysia under the Research University Funding Scheme (Q.J130000.7110.02H47) for supporting the related research.

References

1. Sloman, M., Lupu, E.: Engineering policy-based ubiquitous systems. The Computer Journal, 1113–1127 (2010)
2. Che, K., Li, L.-L., Niu, X.-T., Xing, S.-T.: Research of software development methodology based on self-adaptive multi-agent systems. In: 2009 IEEE International Symposium on IT in Medicine & Education (ITME 2009), vol. 1, pp. 235–240 (2009)
3. Nagib, G., Ali, W.G.: Network routing protocol using Genetic Algorithms. International Journal of Electrical & Computer Sciences 10, 40–44 (2010)
4. Dijkstra, E.W.: A Note on Two Problems in Connection with Graphs. Journal of Numerical Mathematics 1, 269–271 (1959)
5. Arokhlo, M.Z., Selamat, A., Hashim, S.Z.M., Selamat, M.H.: Multi-agent Reinforcement Learning for Route Guidance System. IJACT: InternationalJournal of Advancements in Computing Technology 3(6), 224–232 (2011)
6. Vasirani, M., Ossowski, S.: A market-inspired approach to reservation-based ur- ban road traffic management. In: AAMAS 2009: Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems, Budapest, Hungary, pp. 617–624 (2009)
7. Schulz, F., Wagner, D., Weihe, K.: Dijkstra's algorithm on-line: an empirical case study from public railroad transport. J. Exp. Algorithmics 5, 12 (2000)
8. Andersson, J., de Lemos, R., Malek, S., Weyns, D.: Reflecting on self-adaptive software systems. In: ICSE Workshop on Software Engineering for Adaptive and Self-Managing Systems, Vancouver, Canada, pp. 38–47 (2009)
9. Zhang, J., Goldsby, H.J., Cheng, B.H.: Modular verification of dynamically adaptive systems. In: Proceedings of the 8th ACM International Conference on Aspect-oriented Software Development, AOSD 2009, pp. 161–172. ACM, New York (2009)
10. Herrmann, K., Gero, M., Geihs, K.: Self-management: The solution to complexity or just an- other problem. IEEE Distributed Systems Online 6 (2005)
11. Chen, B., Cheng, H.H., Palen, J.: Integrating mobile agent technology with multi-agent systems for distributed traffic detection and management systems (2009)
12. Zhang, Z., Xu, J.-M.: A dynamic route guidance arithmetic based on reinforcement learning. In: Proceedings of 2005 International Conference on Machine Learning and Cybernetics, China, vol. 6, pp. 3607–3611 (2005)
13. Zou, L., Xu, J.-M., Zhu, L.: Application of genetic algorithm in dynamic route guidance system. Journal of Transportation Systems Engineering and Information Technology 7, 45–48 (2007)